

Homotopies for Free!

Taichi Uemura

January 30, 2017

Abstract

We show “free theorems” in the style of Wadler on homotopy type theory as consequences of relational parametricity. It follows that, for each space, its higher inductive definition and impredicative definition have the same homotopy groups. An impredicative definition of a space can help us to find generators of homotopy groups of the space.

1 Introduction

Given a closed term of type of polymorphic functions defined in homotopy type theory [Uni13], we can derive a theorem that it satisfies. Any closed term of the same type satisfies the same theorem. For example, let t be a closed term of type

$$t : \prod_{X:\mathcal{U}} \prod_{x:X} x = x \rightarrow x = x.$$

Then we have a theorem

$$\prod_{X,X':\mathcal{U}} \prod_{f:X \rightarrow X'} \prod_{x:X} \prod_{p:x=x} t(fp) = f(tp)$$

in homotopy type theory, in the sense that there is a closed term of this type.

Such theorems are “free theorems” in the style of Wadler [Wad89] on homotopy type theory. A difference between free theorems on homotopy type theory and original free theorems on polymorphic type theory is that on homotopy type theory they are represented by *homotopies* instead of *equalities*. This difference causes some problems related to higher dimensional homotopies, but one can solve them by considering relational parametricity for higher dimensional types.

Consider a canonical embedding

$$i : A \rightarrow \prod_{X:\mathcal{U}} (A \rightarrow X) \rightarrow X$$

for a type $A : \mathcal{U}$. On polymorphic type theory it follows from relational parametricity that i is an isomorphism. On homotopy type theory from relational parametricity for the codomain of this type we can derive only the fact that i is *0-connected*, that is, it induces a bijection between the sets of connected components. A 0-connected map is far from an isomorphism. However, from relational parametricity for the type

$$\prod_{X:\mathcal{U}} \prod_{g:A \rightarrow X} \Omega^n(X, ga)$$

for all $n \geq 1$ and $a : A$, one can show that i is *∞ -connected*, that is, it induces a bijection between the n -th homotopy groups for each $n \geq 0$. Hence the types A and $\prod_{X:\mathcal{U}} (A \rightarrow X) \rightarrow X$ are equivalent from homotopical point of view.

For a concrete (higher) inductive type A , this ∞ -connected map and the recursion principle of A give an *impredicative definition* of A which has the same homotopy groups as A and is defined in Martin-Löf type theory [ML75] with universes but without univalence or higher inductive types. For example, an impredicative definition of n -sphere is

$$\prod_{X:\mathcal{U}} \prod_{x:X} \Omega^n(X, x) \rightarrow X.$$

Some of generators of homotopy groups of a space can be defined in an impredicative style without univalence or higher inductive types and implemented in an existing proof assistant without any hacks. It also is possible to let a proof assistant search for generators of homotopy groups of a space.

Free theorems on polymorphic type theory are corollaries of the *relational parametricity theorem*, originally called the *abstraction theorem* by Reynolds [Rey83]. Informally, it says that any function takes related values to related values. Relational parametricity on dependent type theory is studied by Atkey, Ghani and Johann for Martin-Löf type theory [AGJ14], and by Bernardy, Jansson and Paterson for pure type system [BJP12]. We show a similar theorem on homotopy type theory.

The key to prove the relational parametricity theorem is the fact that a type theory of binary type families in homotopy type theory is again a homotopy type theory. There is a category-theoretic proof of this fact using Shulman's inverse diagrams of type-theoretic fibration categories [Shu15] or fibred type-theoretic fibration categories introduced by the author [Uem17]. In this paper we give a syntactic proof in order to make the paper self-contained. We also show a new result on inductive data types: for a type

theory with *indexed W-types*, originally called *general trees* [PS89, NPS90], the type theory of binary type families has indexed W-types.

Organization. We begin in Section 2 by recalling some important types and functions in homotopy type theory. Section 3 and 4 are the core of this paper. In Section 3 we explain what the relational parametricity theorem is. In Section 4, we give some free theorems as corollaries of relational parametricity. In Section 5, we explain how helpful relational parametricity is to give generators of homotopy groups of a space. We prove the relational parametricity theorem in Section 6, 7 and 8.

2 Preliminaries on Homotopy Type Theory

We recall some types and functions in homotopy type theory which are used in Section 3 and 4.

The key idea of homotopy type theory is to identify types as spaces, elements as points and equalities as *paths*. We think of an identity type $x : A, y : A \vdash x = y$ type as the *space of paths from x to y* . Under this identification, reflexivity, transitivity and symmetry correspond to *constant path* $\text{refl}_x : x = x$, *path concatenation* $(-) \cdot (-) : x = y \rightarrow y = z \rightarrow x = z$ and *path inversion* $(-)^{-1} : x = y \rightarrow y = x$ respectively. A function $f : A \rightarrow B$ acts on paths as $\text{ap}(f, -) : x = y \rightarrow fx = fy$ for all $x, y : A$, and we will often write $\text{ap}(f, p)$ as fp for $p : x = y$. Corresponding to indiscernability of identicals, there is a function $\text{transport}^C(p, -) : C(x) \rightarrow C(y)$ for $x : A \vdash C(x)$ type, $x, y : A$ and $p : x = y$.

A function $f : A \rightarrow B$ also acts on higher dimensional paths. For $x_1, y_1 : A, x_2, y_2 : x_1 = y_1, \dots, x_n, y_n : x_{n-1} = y_{n-1}$, we define $\text{ap}_n(f, -) : x_n = y_n \rightarrow \text{ap}_{n-1}(f, x_n) = \text{ap}_{n-1}(f, y_n)$ as $\text{ap}_0(f, z) \equiv fz$ and $\text{ap}_n(f, p) \equiv \text{ap}(\text{ap}_{n-1}(f, -), p)$. We often write $\text{ap}_n(f, p)$ as fp . There are compositions of higher dimensional paths. For $x_0, y_0 : A, x_1, y_1 : x_0 = y_0, \dots, x_n, y_n : x_{n-1} = y_{n-1}, \sigma : x_n = y_n, p : x' = x_0$ and $q : y_0 = y'$, we set $p \cdot_l \sigma \equiv \text{ap}_n(\lambda s. p \cdot s, \sigma)$ and $\sigma \cdot_r q \equiv \text{ap}_n(\lambda s. s \cdot q, \sigma)$. These operations \cdot_l and \cdot_r are called *whiskering*.

A *pointed type* is a pair (A, a) of type A and its inhabitant $a : A$ called a *base point*. For a pointed type (A, a) and a natural number $n \geq 0$, the *n -th loop space* $\Omega^n(A, a)$ of A at a is a pointed type defined inductively as $\Omega^0(A, a) \equiv (A, a)$ and $\Omega^{n+1}(A, a) \equiv \Omega^n(a = a, \text{refl}_a)$. Write $\text{refl}_a^n : \Omega^n(A, a)$ for the base point of $\Omega^n(A, a)$. A function $f : A \rightarrow B$ acts on loop spaces as $\text{ap}_n(f, -) : \Omega^n(A, a) \rightarrow \Omega^n(B, fa)$.

A path space of a product space $A \times B$ is a product of path spaces:

$(\langle a, b \rangle = \langle a', b' \rangle) \simeq (a = a') \times (b = b')$ for $a, a' : A$ and $b, b' : B$. We think of a pair $\langle p, q \rangle$ of paths $p : a = a'$ and $q : b = b'$ as a path $\langle a, b \rangle = \langle a', b' \rangle$ in $A \times B$. Similarly, we regard a pair $\langle l, k \rangle$ of n -loops $l : \Omega^n(A, a)$ and $k : \Omega^n(B, b)$ as an n -loop in $A \times B$ at $\langle a, b \rangle$.

Let $x : A \vdash B(x)$ type be a type family. For a path $p : a = a'$ in A and points $b : B(a)$ and $b' : B(a')$, the *path space from b to b' over p* , written $b =_p b'$, is the type $\text{transport}^B(p, b) = b'$. For an n -loop $l : \Omega^n(A, a)$ and a point $b : B(a)$, the *n -th loop space of B at b over l* , written $\Omega_l^n(B, b)$, is the type $\text{ap}_{n-1}(\lambda p. \text{transport}^B(p, b), l) = \text{refl}_b^{n-1}$.

3 Relational Parametricity Explained

Relational parametricity on polymorphic type theory is explained in terms of set-theoretic relations. On dependent type theory, we use type-theoretic relations, namely binary type families.

For a binary type family $x : A, x' : A' \vdash \mathcal{A}(x, x')$ type, a *family on \mathcal{A}* is a triple of $x : A \vdash B(x)$ type, $x' : A' \vdash B'(x')$ type and $x : A, x' : A', \bar{x} : \mathcal{A}(x, x'), y : B(x), y' : B'(x') \vdash \mathcal{B}(\bar{x}, y, y')$ type, written $\bar{x} : \mathcal{A} \vdash \mathcal{B}(\bar{x})$ rel in short. Note that \mathcal{B} depends on $x : A$ and $x' : A'$ implicitly.

Let $\bar{x} : \mathcal{A} \vdash \mathcal{B}(\bar{x})$ rel be a family on a binary type family \mathcal{A} . The *dependent product of \mathcal{B} over \mathcal{A}* is the binary type family

$$f : \prod_{x:A} B(x), f' : \prod_{x':A'} B'(x') \vdash \prod_{x:A} \prod_{x':A'} \prod_{\bar{x}:\mathcal{A}(x,x')} \mathcal{B}(\bar{x}, fx, f'x') \text{ type.}$$

The *dependent sum of \mathcal{B} over \mathcal{A}* is the binary type family

$$z : \sum_{x:A} B(x), z' : \sum_{x':A'} B'(x') \vdash \sum_{\bar{x}:\mathcal{A}(\text{pr}_1(z), \text{pr}_1(z'))} \mathcal{B}(\bar{x}, \text{pr}_2(z), \text{pr}_2(z')) \text{ type.}$$

For a binary type family \mathcal{A} , the *path space of \mathcal{A}* is the family

$$\begin{aligned} & x_0 : A, x'_0 : A', \bar{x}_0 : \mathcal{A}(x_0, x'_0), x_1 : A, x'_1 : A', \bar{x}_1 : \mathcal{A}(x_1, x'_1), \\ & p : x_0 = x_1, p' : x'_0 = x'_1 \vdash \bar{x}_0 =_{\langle p, p' \rangle} \bar{x}_1 \text{ type} \end{aligned}$$

on two copies of \mathcal{A} .

A *universe* of binary type families is a binary type family

$$X : \mathcal{U}, X' : \mathcal{U} \vdash X \rightarrow X' \rightarrow \mathcal{U} \text{ type}$$

where $\vdash \mathcal{U}$ type is a universe of types.

For each type constant C (for example, $\mathbf{0}$, $\mathbf{1}$, $\mathbf{2}$, \mathbb{N} , \mathbb{S}^1 , \mathbb{S}^2 and so on), we associate it with a binary type family

$$c : C, c' : C \vdash c = c' \text{ type.}$$

Then, by induction, we can associate each type family $x : X \vdash A(x)$ type with a family of binary type families

$$x : X, x' : X, \bar{x} : R[X](x, x'), a : A(x), a' : A(x') \vdash R[A](\bar{x}, a, a') \text{ type.}$$

Now relational parametricity can be described as follows:

Theorem (Relational Parametricity). *For each term $x : X \vdash t(x) : A(x)$, there exists a term*

$$x : X, x' : X', \bar{x} : R[X](x, x') \vdash \hat{t}(\bar{x}) : R[A](\bar{x}, t(x), t(x')).$$

In particular, for each closed term $\vdash t : A$, there exists a closed term

$$\vdash \hat{t} : R[A](t, t).$$

4 Relational Parametricity Applied

4.1 Concatenation of a Loop

Let t be a closed term of type

$$t : \prod_{X:\mathcal{U}} \prod_{x:X} x = x \rightarrow x = x.$$

One might guess that t is an iterated concatenation of a loop, that is,

$$t(p) \equiv \underbrace{p \cdot \dots \cdot p}_{n \text{ times}}$$

for a fixed integer n , where negative n means $(-n)$ times concatenation of the inversion of p . One might be able to find another function, but as long as t is a closed term of this type, we can derive a theorem that t satisfies. We show that the type

$$\prod_{X, X':\mathcal{U}} \prod_{f:X \rightarrow X'} \prod_{x:X} \prod_{p:x=p} t(fp) = f(tp)$$

is inhabited. From relational parametricity we have a closed term

$$\hat{t} : \prod_{(X:\mathcal{U}, X':\mathcal{U}, \mathcal{X}:X \rightarrow X' \rightarrow \mathcal{U})} \prod_{(x:X, x':X', \bar{x}:\mathcal{X}(x, x'))} \prod_{(p:x=x, p':x'=x', \bar{p}:\bar{x}=\langle p, p' \rangle \bar{x})} \bar{x} =_{\langle tp, tp' \rangle} \bar{x}.$$

For a function $f : X \rightarrow X'$ of \mathcal{U} -small types, let $\mathcal{X}(x, x') \equiv fx = x'$. One can prove that, for $p : x = x$, $p' : x' = x'$ and $\bar{x} : fx = x'$, the type $\bar{x} =_{\langle p, p' \rangle} \bar{x}$ is equivalent to the type $\bar{x} \cdot p' = fp \cdot \bar{x}$. Letting $x' \equiv fx$ and $\bar{x} \equiv \text{refl}_{fx}$, we have an inhabitant of the type

$$\prod_{(p:x=x, p':fx=fx, \bar{p}:p'=fp)} tp' = f(tp).$$

Finally we set $p' \equiv fp$ and $\bar{p} \equiv \text{refl}_{fp}$. Then we have an inhabitant of the type

$$\prod_{x:X} \prod_{p:x=x} t(fp) = f(tp).$$

4.2 Loop Operations

The example in Section 4.1 can be generalized. Let n and k be natural numbers and t a closed term of type

$$t : \prod_{X:\mathcal{U}} \prod_{x:X} \Omega^n(X, x) \rightarrow \Omega^k(X, x).$$

The example in Section 4.1 is the case where $n = k = 1$. This type represents the k -th loop space of n -sphere, discussed in Section 5, and thus we could not guess what function t is. However, we can derive a theorem about t . We show that the type

$$\prod_{X, X':\mathcal{U}} \prod_{f:X \rightarrow X'} \prod_{x:X} \prod_{p:\Omega^n(X, x)} t(fp) = f(tp)$$

is inhabited. From relational parametricity we have a closed term

$$\hat{t} : \prod_{(X:\mathcal{U}, X':\mathcal{U}, \mathcal{X}:X \rightarrow X' \rightarrow \mathcal{U})} \prod_{(x:X, x':X', \bar{x}:\mathcal{X}(x, x'))} \prod_{(p:\Omega^n(X, x), p':\Omega^n(X, x'), \bar{p}:\Omega^n_{\langle p, p' \rangle}(X, \bar{x}))} \Omega^k_{\langle tp, tp' \rangle}(\mathcal{X}, \bar{x}).$$

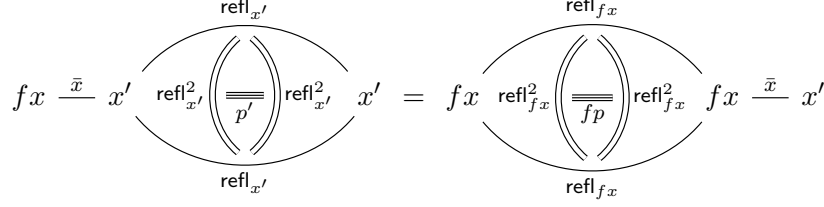


Figure 1: The type $\bar{x} \cdot_1 p' = fp \cdot_r \bar{x}$ in case $n = 3$

For a function $f : X \rightarrow X'$ of \mathcal{U} -small types, let $\mathcal{X}(x, x') \equiv fx = x'$. One can prove that, for $p : \Omega^n(X, x)$, $p' : \Omega^n(X, x')$ and $\bar{x} : fx = x'$, the type $\Omega_{\langle p, p' \rangle}^n(\mathcal{X}, \bar{x})$ is equivalent to the type $\bar{x} \cdot_1 p' = fp \cdot_r \bar{x}$. Letting $x' \equiv fx$, $\bar{x} \equiv \text{refl}_{fx}$, $p' \equiv fp$ and $\bar{p} \equiv \text{refl}_{fp}$, we have an inhabitant of the type

$$\prod_{x:X} \prod_{p:\Omega^n(X,x)} t(fp) = f(tp).$$

4.3 Action on Loops

Let t be a closed term of type

$$t : \prod_{X,Y:\mathcal{U}} \prod_{f:X \rightarrow Y} \prod_{x:X} x = x \rightarrow fx = fx.$$

One might guess that $t(f, p) \equiv \text{ap}(f, p)$. Of course, t could be another function, for example, $t(f, p) \equiv \text{ap}(f, p \cdot p)$. However, intuitively only $\text{ap}(f, p)$ is an interesting function of this type, because $\text{ap}(f, p \cdot p)$ is a composition of $\text{ap}(f, p)$ and a loop concatenation, and the latter does not use f .

Let us to formulate this intuition. We show that the type

$$\prod_{X,Y:\mathcal{U}} \prod_{f:X \rightarrow Y} \prod_{x:X} \prod_{p:x=x} t(f, p) = f(t(\text{id}_X, p))$$

is inhabited. This means that, for any t , $t(f, -)$ is a composition of $\text{ap}(f, -)$ after a loop operation $t(\text{id}_X, -) : x = x \rightarrow x = x$. From relational parametricity we have an inhabitant of the type

$$\prod_{X', X, Y', Y:\mathcal{U}} \prod_{g:X' \rightarrow X} \prod_{h:Y' \rightarrow Y} \prod_{f':X' \rightarrow Y'} \prod_{f:X \rightarrow Y} \prod_{\sigma:\prod_{x':X'} f(gx') = h(f'x')} \prod_{x':X'} \prod_{p':x'=x'} t(f, gp') \cdot \sigma(x') = \sigma(x') \cdot h(t(f', p')).$$

Letting $X' \equiv Y' \equiv X$, $h \equiv f$, $f' \equiv g \equiv \text{id}_X$ and $\sigma \equiv \lambda x. \text{refl}_{f_x}$, we have

$$\prod_{X,Y:\mathcal{U}} \prod_{f:X \rightarrow Y} \prod_{x:X} \prod_{p:x=x} t(f,p) = f(t(\text{id}_X, p)).$$

Note that, from Section 4.1, we also have $f(t(\text{id}_X, p)) = t(\text{id}_Y, fp)$.

4.4 An Embedding

For a type $A : \mathcal{U}$, let $\tilde{A} \equiv \prod_{X:\mathcal{U}} (A \rightarrow X) \rightarrow X$. There are back and forth functions between the types A and \tilde{A} as follows:

$$\begin{aligned} i &: A \rightarrow \prod_{X:\mathcal{U}} (A \rightarrow X) \rightarrow X \\ i &\equiv \lambda(a : A). \lambda(X : \mathcal{U}, g : A \rightarrow X). ga \\ j &: (\prod_{X:\mathcal{U}} (A \rightarrow X) \rightarrow X) \rightarrow A \\ j &\equiv \lambda(\varphi : \prod_{X:\mathcal{U}} (A \rightarrow X) \rightarrow X). \varphi_A(\text{id}_A). \end{aligned}$$

Clearly $j \circ i \equiv \text{id}_A$, while $i \circ j \equiv \text{id}_{\tilde{A}}$ or even $i \circ j \sim \text{id}_{\tilde{A}}$ does not hold. However, given a closed term $t : \tilde{A}$, we can construct a closed term of type

$$\prod_{X:\mathcal{U}} \prod_{g:A \rightarrow X} (i(jt))g = tg.$$

To show this, let $t : \tilde{A}$ be a closed term. From relational parametricity we can get a closed term of type

$$\prod_{X_0, X:\mathcal{U}} \prod_{f:X_0 \rightarrow X} \prod_{g:A \rightarrow X_0} f(tg) = t(f \circ g).$$

If $X_0 \equiv A$, letting $g \equiv \text{id}_A$ we get an inhabitant of the type $f(t(\text{id}_A)) = t(f)$. Now, for $X : \mathcal{U}$ and $g : A \rightarrow X$, we have $(i(jt))g = g(jt) = g(t(\text{id}_A)) = t(g)$.

4.5 An ∞ -Connected Map

For a type A , let $\pi_0(A)$ be the set of homotopy equivalence classes of closed terms of A which we call the *0-th homotopy group of A* . For a point $a : A$ and a natural number n , the *n -th homotopy group of A at a* , written $\pi_n(A, a)$, is the set $\pi_0(\Omega^n(A, a))$. From Section 4.4, we get a bijection $\pi_0(A) \rightarrow \pi_0(\tilde{A})$. We can extend this result to all homotopy groups. We show that $i : A \rightarrow \tilde{A}$

is ∞ -connected in the sense that it induces a bijection between the n -th homotopy groups for each $n \geq 0$.

For a pointed types $(A, a) : \mathcal{U}_\bullet$, \tilde{A} has a base point $\tilde{a} \equiv \lambda(X : \mathcal{U}, g : A \rightarrow X).ga$. The maps i and j preserve base points, and thus they induce maps $\Omega^n(i) : \Omega^n(A, a) \rightarrow \Omega^n(\tilde{A}, \tilde{a})$ and $\Omega^n(j) : \Omega^n(\tilde{A}, \tilde{a}) \rightarrow \Omega^n(A, a)$. Identifying $\Omega^n(\tilde{A}, \tilde{a})$ with $\prod_{X:\mathcal{U}} \prod_{g:A \rightarrow X} \Omega^n(X, ga)$, we get:

$$\begin{aligned} \Omega^n(i) : \Omega^n(A, a) &\rightarrow \prod_{X:\mathcal{U}} \prod_{g:A \rightarrow X} \Omega^n(X, ga) \\ \Omega^n(i) &= \lambda(p : \Omega^n(A, a)).\lambda(X : \mathcal{U}, g : A \rightarrow X).gp \\ \Omega^n(j) : (\prod_{X:\mathcal{U}, g:A \rightarrow X} \Omega^n(X, ga)) &\rightarrow \Omega^n(A, a) \\ \Omega^n(j) &= \lambda(\varphi : \prod_{X:\mathcal{U}} \prod_{g:A \rightarrow X} \Omega^n(X, ga)).\varphi_A(\text{id}_A). \end{aligned}$$

Then we have $\Omega^n(j) \circ \Omega^n(i) = \text{id}_{\Omega^n(A, a)}$. For a closed term $t : \Omega^n(\tilde{A}, \tilde{a})$, we can construct a closed term of type

$$\prod_{X:\mathcal{U}} \prod_{g:A \rightarrow X} (\Omega^n(i)(\Omega^n(j)t))g = tg$$

in a similar way to Section 4.4. Thus we conclude that the map i induces a bijection

$$\pi_n(A, a) \rightarrow \pi_n(\tilde{A}, \tilde{a})$$

for each $n \geq 0$.

5 Homotopy Groups

From Section 4.5, for each type $A : \mathcal{U}$ we have an ∞ -connected map $i : A \rightarrow \tilde{A}$ where $\tilde{A} \equiv \prod_{X:\mathcal{U}} (A \rightarrow X) \rightarrow X$. For a concrete (higher) inductive type A , using the recursion principle of A we have an *impredicative definition* of A which can be defined without univalence or higher inductive types. If A has a base point $a_0 : A$, an impredicative definition of A is of the form

$$\prod_{X:\mathcal{U}} \prod_{x:X} F_A(X, x) \rightarrow X$$

and its n -th loop space is

$$\prod_{X:\mathcal{U}} \prod_{x:X} F_A(X, x) \rightarrow \Omega^n(X, x),$$

where $F_A(X, x)$ is a type defined from X and x using only dependent products, dependent sums and path spaces. An impredicative definition of a type A suggests that we can *construct* generators of homotopy groups of A without univalence or higher inductive types, although we need univalence and higher inductive types to *prove* that they are actually generators of homotopy groups.

In this section we describe impredicative definitions of some higher inductive types. We also define a Hopf fibration and give a generator of the third homotopy group of 2-sphere in an impredicative style.

5.1 Pushouts

Let $f : C \rightarrow A$ and $g : C \rightarrow B$ be functions. The *pushout* $A \amalg_C B$ of f and g is a higher inductive type generated by two point constructors $\text{inl} : A \rightarrow A \amalg_C B$ and $\text{inr} : B \rightarrow A \amalg_C B$ and a path constructor $\text{glue} : \prod_{z:C} \text{inl}(fz) = \text{inr}(gz)$. It has a recursion principle

$$A \amalg_C B \rightarrow X \simeq \sum_{s:A \rightarrow X} \sum_{t:B \rightarrow X} \prod_{z:C} s(fz) = t(gz).$$

Therefore

$$\widetilde{A \amalg_C B} \simeq \prod_{X:\mathcal{U}} \prod_{s:A \rightarrow X} \prod_{t:B \rightarrow X} \left(\prod_{z:C} s(fz) = t(gz) \right) \rightarrow X.$$

5.2 Suspensions

For a type A , the *suspension* ΣA of A is the pushout of two copies of the unique map $A \rightarrow \mathbf{1}$. Therefore

$$\widetilde{\Sigma A} \simeq \prod_{X:\mathcal{U}} \prod_{x,y:X} (A \rightarrow x = y) \rightarrow X.$$

We set $\mathbf{N} \equiv \text{inl}(\ast)$ and $\mathbf{S} \equiv \text{inr}(\ast)$ where $\ast : \mathbf{1}$ is the constructor of $\mathbf{1}$.

5.3 Joins

For types A and B , the *join* $A \star B$ of A and B is the pushout of projections $A \times B \rightarrow A$ and $A \times B \rightarrow B$. Therefore

$$\widetilde{A \star B} \simeq \prod_{X:\mathcal{U}} \prod_{s:A \rightarrow X} \prod_{t:B \rightarrow X} \left(\prod_{a:A} \prod_{b:B} sa = tb \right) \rightarrow X.$$

5.4 Spheres

For a natural number n , the n -sphere \mathbb{S}^n is a higher inductive type generated by a point constructor $\text{base}_n : \mathbb{S}^n$ and a path constructor $\text{loop}_n : \Omega^n(\mathbb{S}^n, \text{base}_n)$. It has a recursion principle

$$\mathbb{S}^n \rightarrow X \simeq \sum_{x:X} \Omega^n(X, x).$$

Therefore

$$\widetilde{\mathbb{S}}^n \simeq \prod_{X:\mathcal{U}} \prod_{x:X} \Omega^n(X, x) \rightarrow X.$$

Recall that n -spheres \mathbb{S}^n are also defined inductively as

$$\begin{aligned} \mathbb{S}^0 &\equiv \mathbf{2} \\ \mathbb{S}^{n+1} &\equiv \Sigma \mathbb{S}^n. \end{aligned}$$

Thus

$$\widetilde{\mathbb{S}}^n \simeq \prod_{X:\mathcal{U}} \prod_{x_0, y_0 : X} \prod_{x_1, y_1 : x_0 = y_0} \cdots \prod_{x_n, y_n : x_{n-1} = y_{n-1}} X.$$

5.5 A Hopf Fibration

A *Hopf fibration* is a function $\mathbb{S}^3 \rightarrow \mathbb{S}^2$ whose fiber at the base point is \mathbb{S}^1 . Identifying $\mathbb{S}^3 \simeq \mathbb{S}^1 \star \mathbb{S}^1$ and $\mathbb{S}^2 \simeq \Sigma \mathbb{S}^1$, a Hopf fibration $h : \mathbb{S}^1 \star \mathbb{S}^1 \rightarrow \Sigma \mathbb{S}^1$ is defined as

$$\begin{aligned} h(\text{inl}(x)) &\equiv \text{N} \\ h(\text{inr}(y)) &\equiv \text{S} \\ h(\text{glue}(x, y)) &= \text{glue}(h_1(x, y)), \end{aligned}$$

where

$$\begin{aligned} h_1 : \mathbb{S}^1 &\rightarrow \mathbb{S}^1 \rightarrow \mathbb{S}^1 \\ h_1(\text{base}_1, y) &\equiv y \\ h_1(\text{loop}_1, \text{base}_1) &= \text{loop}_1^{-1} \end{aligned}$$

and $h_1(\text{loop}_1, \text{loop}_1)$ is given by a proof of $\text{loop}_1^{-1} \cdot \text{loop}_1 = \text{loop}_1 \cdot \text{loop}_1^{-1}$.

We define the Hopf fibration in an impredicative style. Observe that

$$\begin{aligned} \mathbb{S}^1 \star \mathbb{S}^1 \rightarrow X &\simeq \sum_{f, g : \mathbb{S}^1 \rightarrow X} \prod_{x, y : \mathbb{S}^1} fx = gy \\ &\simeq \sum_{x:X} \sum_{l:x=x} \sum_{y:X} \sum_{k:y=y} \sum_{p:x=y} \sum_{\alpha:l \cdot p=p} \sum_{\beta:p \cdot k=p} (\alpha \cdot_r k) \cdot \beta = (l \cdot_l \beta) \cdot \alpha \end{aligned}$$

$$\begin{array}{ccc}
x & \xrightarrow{p} & y \\
\downarrow l & \begin{array}{c} \parallel \alpha \\ \nearrow p \\ \parallel \beta \end{array} & \downarrow k \\
x & \xrightarrow{p} & y
\end{array}
=
\begin{array}{ccc}
x & \xrightarrow{p} & y \\
\downarrow l & \begin{array}{c} \searrow p \\ \parallel \alpha \\ \parallel \beta \end{array} & \downarrow k \\
x & \xrightarrow{p} & y
\end{array}$$

Figure 2: The type $(\alpha \cdot_r k) \cdot \beta = (l \cdot_l \beta) \cdot \alpha$

and

$$\begin{aligned}
\Sigma \mathbb{S}^1 \rightarrow X &\simeq \sum_{x,y:X} \mathbb{S}^1 \rightarrow x = y \\
&\simeq \sum_{x,y:X} \sum_{p:x=y} p = p.
\end{aligned}$$

Then we define

$$\begin{aligned}
h : & \left(\prod_{X:\mathcal{U}} \prod_{x,y:X} \prod_{l:x=x} \prod_{k:y=y} \prod_{p:x=y} \prod_{\alpha:l \cdot p = p} \prod_{\beta:p \cdot k = p} (\alpha \cdot_r k) \cdot \beta = (l \cdot_l \beta) \cdot \alpha \rightarrow X \right) \\
& \rightarrow \left(\prod_{X:\mathcal{U}} \prod_{x,y:X} \prod_{p:x=y} p = p \rightarrow X \right)
\end{aligned}$$

$$h(f) \equiv \lambda(X, x, y, p, \sigma). f(X, x, y, \text{refl}_x, \text{refl}_y, p, \sigma^{-1}, \sigma, \tilde{\sigma})$$

where $\tilde{\sigma}$ is a proof of $\sigma^{-1} \cdot \sigma = \sigma \cdot \sigma^{-1}$. One can see that this impredicative definition is essentially same as the inductive definition.

5.6 A Generator of $\pi_3(\mathbb{S}^2)$

The impredicative definition of a Hopf fibration gives us a candidate for a generator of $\pi_3(\mathbb{S}^2)$.

First we define a 3-loop of $\mathbb{S}^1 \star \mathbb{S}^1$ in an impredicative style. We have to construct a function

$$\begin{aligned}
l_3 : & \prod_{X:\mathcal{U}} \prod_{x,y:X} \prod_{l:x=x} \prod_{k:y=y} \prod_{p:x=y} \prod_{\alpha:l \cdot p = p} \prod_{\beta:p \cdot k = p} \\
& (\alpha \cdot_r k) \cdot \beta = (l \cdot_l \beta) \cdot \alpha \rightarrow \Omega^3(X, x).
\end{aligned}$$

By path induction on p , we can assume $y \equiv x$ and $p \equiv \text{refl}_x$. Then the goal becomes

$$\prod_{X:\mathcal{U}} \prod_{x:X} \prod_{l,k:x=x} \prod_{\alpha:l=\text{refl}_x} \prod_{\beta:k=\text{refl}_x} (\alpha \cdot_r k) \cdot \beta = (l \cdot_l \beta) \cdot \alpha \rightarrow \Omega^3(X, x).$$

By path induction on α and β , we can assume $l \equiv k \equiv \text{refl}_x$ and $\alpha \equiv \beta \equiv \text{refl}_x^2$. Then the goal becomes

$$\prod_{X:\mathcal{U}} \prod_{x:X} \Omega^3(X, x) \rightarrow \Omega^3(X, x),$$

and thus give the identity function.

Now we can define a 3-loop of $\Sigma\mathbb{S}^1$ in a similar way to the Hopf fibration:

$$\begin{aligned} c : \prod_{X:\mathcal{U}} \prod_{x,y:X} \prod_{p:x=y} p = p &\rightarrow \Omega^3(X, x) \\ c &\equiv \lambda(X, x, y, p, \sigma). l_3(X, x, y, \text{refl}_x, \text{refl}_y, p, \sigma^{-1}, \sigma, \tilde{\sigma}). \end{aligned}$$

Here is a natural question.

Question 5.1. *Is any generator of a homotopy group of a space definable in an impredicative style without univalence or higher inductive types?*

This question is important because it measures power of univalence and higher inductive types. If the answer to the question is yes, we can say, informally, that univalence and higher inductive types give proofs that some elements are different but do not generate new elements, although there is a problem which terms we should think of as proofs, because in dependent type theory elements and proofs are not distinguished.

6 Homotopy Type Theory

In the rest of this paper we prove the relational parametricity theorem. We begin with a quick review of homotopy type theory [Uni13].

In this paper we consider the Martin-Löf's dependent type theory \mathbb{T} with countably many univalent universes

$$\mathcal{U}_0 : \mathcal{U}_1 : \mathcal{U}_2 : \dots,$$

an empty type $\mathbf{0} : \mathcal{U}_0$, a one point type $\mathbf{1} : \mathcal{U}_0$, a two point type $\mathbf{2} : \mathcal{U}_0$, indexed \mathbf{W} -types $\mathbf{W}[t, A, B]$ and n -spheres $\mathbb{S}^n : \mathcal{U}_0$. Existence of \mathbf{W} -types is not enough to construct \mathbf{W} -types in a relational model, and we require *indexed \mathbf{W} -types*. For the same reason, some class of indexed higher inductive types is needed to construct general higher inductive types in a relational model, but we do not know such a class of higher inductive types. Therefore we deal with only constant higher inductive types \mathbb{S}^n .

For a type family $i : I, x : A(i) \vdash B(x)$ **type** and a function $i : I, x : A(i), y : B(x) \vdash t(y) : I$, the **W-type** $W[t, A, B]$ of B on A indexed over t is an inductive type family $i : I \vdash W[t, A, B](i)$ **type** with a single constructor

$$i : I, a : A(i), f : \prod_{y:B(a)} W[t, A, B](ty) \vdash \text{sup}_{[t, A, B]}(a, f) : W[t, A, B](i).$$

We often omit the subscript $_{[t, A, B]}$ of the constructor and write it simply as **sup**. The indexed W-type has an induction principle: given a type family $i : I, w : W[t, A, B](i) \vdash D(w)$ **type** and a term

$$\begin{aligned} i : I, a : A(i), f : \prod_{y:B(a)} W[t, A, B](ty), \\ g : \prod_{y:B(a)} D(fy) \vdash d(a, f, g) : D(\text{sup}(a, f)), \end{aligned}$$

we get a term

$$i : I, w : W[t, A, B](i) \vdash \text{ind}_{W[t, A, B]}^D(d, w) : D(w)$$

together with a computational rule

$$\text{ind}_{W[t, A, B]}^D(d, \text{sup}(a, f)) \equiv d(a, f, \lambda(y : B(a)). \text{ind}_{W[t, A, B]}^D(d, fy)).$$

There are projections

$$\begin{aligned} i : I \vdash \text{pr}_1 : W[t, A, B](i) &\rightarrow A(i) \\ i : I \vdash \text{pr}_1(\text{sup}(a, f)) &\equiv a \\ i : I \vdash \text{pr}_2 : \prod_{w:W[t, A, B](i)} \prod_{y:B(\text{pr}_1(w))} &W[t, A, B](ty) \\ i : I \vdash \text{pr}_2(\text{sup}(a, f)) &\equiv f \end{aligned}$$

Some important types are definable from these types. Ordinary W-types $W_{x:A}B(x)$ are W-types indexed over the function $B \rightarrow \mathbf{1}$. The type \mathbb{N} of natural numbers is defined as

$$\mathbb{N} \equiv W_{x:\mathbf{2}} \text{rec}_2(\mathbf{0}, \mathbf{1}, x),$$

where $\text{rec}_2(\mathbf{0}, \mathbf{1}) : \mathbf{2} \rightarrow \mathcal{U}_0$ is a function defined by recursion as $\text{rec}_2(\mathbf{0}, \mathbf{1}, \mathbf{0}_2) \equiv \mathbf{0}$ and $\text{rec}_2(\mathbf{0}, \mathbf{1}, \mathbf{1}_2) \equiv \mathbf{1}$. A coproduct $A + B$ of two types $A, B : \mathcal{U}$ is defined as

$$A + B \equiv \sum_{x:\mathbf{2}} \text{rec}_2(A, B, x).$$

7 Relational Model

The key to prove the relational parametricity is the fact that binary type families $x : A, x' : A' \vdash \mathcal{A}(x, x')$ type form again a homotopy type theory. Formally, we construct a type theory $\mathcal{R}el(\mathbb{T})$ where types are binary type families.

Families $\bar{x} : \mathcal{A} \vdash \mathcal{B}(\bar{x})$ rel of binary type families are defined in Section 3. A *term* of a family $\bar{x} : \mathcal{A} \vdash \mathcal{B}(\bar{x})$ rel is a triple of terms $x : A \vdash b(x) : B(x)$, $x' : A' \vdash b'(x') : B'(x')$ and $x : A, x' : A', \bar{x} : \mathcal{A}(x, x') \vdash \bar{b}(\bar{x}) : \mathcal{B}(\bar{x}, b, b')$, written $\bar{x} : \mathcal{A} \vdash \bar{b}(\bar{x}) : \mathcal{B}(\bar{x})$ in short. In Section 3, we defined dependent products, dependent sums, path spaces and universes of binary type families. It remains to construct other types and check the univalence axiom.

For a type constant $C \equiv \mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbb{S}^n$, the binary type family $c : C, c' : C \vdash c = c'$ type has the same constructors and satisfies the same induction principle as those of C . For example, $c : \mathbf{2}, c' : \mathbf{2} \vdash c = c'$ type has two constructors $(0_2, 0_2, \text{refl}_{0_2})$ and $(1_2, 1_2, \text{refl}_{1_2})$. To see the induction principle of two point type, let $c : \mathbf{2}, c' : \mathbf{2}, \bar{c} : c = c', x : A(c), x' : A'(c') \vdash \mathcal{A}(\bar{c}, x, x')$ type be a family on $c = c'$ and $(a_0 : A(0_2), a'_0 : A'(0_2), \bar{a}_0 : \mathcal{A}(\text{refl}_{0_2}, a_0, a'_0))$ and $(a_1 : A(1_2), a'_1 : A'(1_2), \bar{a}_1 : \mathcal{A}(\text{refl}_{1_2}, a_1, a'_1))$ be elements of \mathcal{A} . We have to construct terms $c : \mathbf{2} \vdash f(c) : A(c)$, $c' : \mathbf{2} \vdash f'(c') : A'(c')$ and $c : \mathbf{2}, c' : \mathbf{2}, \bar{c} : c = c' \vdash \bar{f}(\bar{c}) : \mathcal{A}(\bar{c}, f(c), f'(c'))$ such that $f(0_2) \equiv a_0$, $f(1_2) \equiv a_1$, $f'(0_2) \equiv a'_0$, $f'(1_2) \equiv a'_1$, $\bar{f}(\text{refl}_{0_2}) \equiv \bar{a}_0$ and $\bar{f}(\text{refl}_{1_2}) \equiv \bar{a}_1$. Define f and f' by $\mathbf{2}$ -induction. By path induction, to construct \bar{f} it suffices to give a term $c : \mathbf{2} \vdash \bar{f}(\text{refl}_c) : \mathcal{A}(\text{refl}_c, f(c), f'(c))$, which is given by $\mathbf{2}$ -induction.

To define indexed W-types, suppose that we get a family of binary type families $\bar{i} : \mathcal{I}, \bar{x} : \mathcal{A}(\bar{i}) \vdash \mathcal{B}(\bar{x})$ rel and a term $\bar{i} : \mathcal{I}, \bar{x} : \mathcal{A}(\bar{i}), \bar{y} : \mathcal{B}(\bar{x}) \vdash \bar{t}(\bar{y}) : \mathcal{I}$. First we have indexed W-types $i : I \vdash W[t, A, B](i)$ type and $i' : I' \vdash W[t, A', B'](i')$ type which we refer to as $W(i)$ and $W'(i')$ respectively. We have to construct a type $i : I, i' : I', \bar{i} : \mathcal{I}(i, i'), w : W(i), w' : W'(i') \vdash \mathcal{W}(\bar{i}, w, w')$ type. Let $J \equiv \sum_{i:I} \sum_{i':I'} \mathcal{I}(i, i') \times W(i) \times W'(i')$. Define type families $j : J \vdash \check{A}(j)$ type and $j : J, x : \check{A}(j) \vdash \check{B}(x)$ type as

$$\begin{aligned} \check{A}(\bar{i}, w, w') &\equiv \mathcal{A}(\bar{i}, \text{pr}_1(w), \text{pr}_1(w')) \\ \check{B}(\bar{i}, w, w'), \bar{a} &\equiv \sum_{b:B(\text{pr}_1(w))} \sum_{b':B'(\text{pr}_1(w'))} \mathcal{B}(\bar{a}, b, b'). \end{aligned}$$

Define a term $j : J, x : \check{A}(j), y : \check{B}(x) \vdash \check{t}(y) : J$ as

$$\check{t}(\bar{i}, w, w'), \bar{x}, (b, b', \bar{b}) \equiv (t(b), t'(b'), \bar{t}(\bar{b}), \text{pr}_2(w)(b), \text{pr}_2(w')(b')).$$

Then we set

$$\mathcal{W}(\bar{i}, w, w') \equiv W[\check{t}, \check{A}, \check{B}](\bar{i}, w, w').$$

We have a constructor

$$\begin{aligned} i : I, i' : I', \bar{i} : \mathcal{I}(i, i'), a : A, a' : A', \bar{a} : \mathcal{A}(\bar{i}, a, a'), \\ f : \prod_{y:B(a)} W(t(y)), f' : \prod_{y':B'(a')} W'(t'(y')), \\ \bar{f} : \prod_{y:B(a)} \prod_{y':B'(a')} \prod_{\bar{y}:\mathcal{B}(\bar{a}, y, y')} \mathcal{W}(\bar{t}(\bar{y}), f(y), f'(y')) \\ \vdash \sup_{[\check{t}, \check{A}, \check{B}]}(\bar{a}, \bar{f}) : \mathcal{W}(\bar{i}, \sup_{[t, A, B]}(a, f), \sup_{[t', A', B']} (a', f')). \end{aligned}$$

One can check the induction principle of indexed W-type, which is left to the reader.

We give a sketch of a proof that a universe $X : \mathcal{U}, X' : \mathcal{U} \vdash X \rightarrow X' \rightarrow \mathcal{U}$ type of binary type families satisfies the univalence axiom. Recall that \mathcal{U} satisfies the univalence axiom if and only if the canonical function

$$\begin{aligned} e : \mathcal{U} \rightarrow \sum_{X, X' : \mathcal{U}} X \simeq X' \\ e(X) \equiv (X, X, \text{id}_X) \end{aligned}$$

is an equivalence. Observe that, in the type theory $\mathcal{R}el(\mathbb{T})$, a function $\bar{f} : \mathcal{A} \rightarrow \mathcal{B}$ is an equivalence if and only if $f : A \rightarrow B$ and $f' : A' \rightarrow B'$ are equivalences and $\bar{f}(x, x') : \mathcal{A}(x, x') \rightarrow \mathcal{B}(f x, f' x')$ is an equivalence for all $x : A$ and $x' : A'$. Therefore, to show that $X \rightarrow X' \rightarrow \mathcal{U}$ is univalent, it suffices to see that

$$\bar{e} : (X \rightarrow X' \rightarrow \mathcal{U}) \rightarrow \sum_{\mathcal{X}, \mathcal{Y} : X \rightarrow X' \rightarrow \mathcal{U}} \prod_{x : X} \prod_{x' : X'} \mathcal{X}(x, x') \simeq \mathcal{Y}(x, x')$$

is an equivalence for all $X, X' : \mathcal{U}$. The codomain of \bar{e} is equivalent to

$$X \rightarrow X' \rightarrow \sum_{\mathcal{X}, \mathcal{Y} : \mathcal{U}} \mathcal{X} \simeq \mathcal{Y},$$

and \bar{e} is homotopic to

$$(X \rightarrow X' \rightarrow e) : (X \rightarrow X' \rightarrow \mathcal{U}) \rightarrow (X \rightarrow X' \rightarrow \sum_{\mathcal{X}, \mathcal{Y} : \mathcal{U}} \mathcal{X} \simeq \mathcal{Y})$$

along this equivalence. The function $(X \rightarrow X' \rightarrow e)$ is an equivalence by univalence of \mathcal{U} .

8 The Relational Parametricity Theorem

In Section 7 we see that the type theory $\mathcal{Rel}(\mathbb{T})$ of binary type families form a Martin-Löf's dependent type theory with countable univalent universes, an empty type, a one point type, a two point type, indexed W-type and n -spheres. Thus we have an interpretation $R : \mathbb{T} \rightarrow \mathcal{Rel}(\mathbb{T})$. R takes a type judgment $x : X \vdash A(x)$ type to a type judgment

$$x : X, x' : X, \bar{x} : R[X], a : A(x), a' : A(x') \vdash R[A](\bar{x}, a, a') \text{ type}$$

and a term judgment $x : X \vdash t(x) : A(x)$ to a term judgment

$$x : X, x' : X, \bar{x} : R[X] \vdash R[t](\bar{x}) : R[A](\bar{x}, t(x), t(x')).$$

Now the relational parametricity theorem is proved by taking $\hat{t} \equiv R[t]$.

Theorem 8.1 (Relational Parametricity). *For each term $x : X \vdash t(x) : A(x)$, there exists a term*

$$x : X, x' : X', \bar{x} : R[X](x, x') \vdash \hat{t}(\bar{x}) : R[A](\bar{x}, t(x), t(x')).$$

References

- [AGJ14] Robert Atkey, Neil Ghani, and Patricia Johann. A relationally parametric model of dependent type theory. In *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '14, pages 503–515, New York, NY, USA, 2014. ACM. doi:10.1145/2535838.2535852.
- [BJP12] Jean-Philippe Bernardy, Patrik Jansson, and Ross Paterson. Proofs for free: Parametricity for dependent types. *Journal of Functional Programming*, 22(2):107–152, 003 2012. doi:10.1017/S0956796812000056.
- [ML75] Per Martin-Löf. An intuitionistic theory of types: Predicative part. In H.E. Rose and J.C. Shepherdson, editors, *Logic Colloquium '73 Proceedings of the Logic Colloquium*, volume 80 of *Studies in Logic and the Foundations of Mathematics*, pages 73 – 118. Elsevier, 1975. doi:10.1016/S0049-237X(08)71945-1.
- [NPS90] Bengt Nordström, Kent Petersson, and Jan M. Smith. *Programming in Martin-Löf's type theory: An Introduction*. Oxford University Press, 1990. URL: <http://www.cse.chalmers.se/research/group/logic/book/>.

- [PS89] Kent Petersson and Dan Synek. A set constructor for inductive sets in Martin-Löf’s type theory. In *Category Theory and Computer Science*, pages 128–140, London, UK, 1989. Springer-Verlag.
- [Rey83] John C. Reynolds. Types, abstraction, and parametric polymorphism. In R.E.A. Mason, editor, *Information Processing ’83*, pages 513–523. North-Holland, 1983.
- [Shu15] Michael Shulman. Univalence for inverse diagrams and homotopy canonicity. *Mathematical Structures in Computer Science*, 25(05):1203–1277, 2015. [arXiv:1203.3253](#), [doi:10.1017/s0960129514000565](#).
- [Uem17] Taichi Uemura. Fibred fibration categories, January 2017. URL: <http://arxiv.org/abs/1602.08206v4>, [arXiv:1602.08206v4](#).
- [Uni13] Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study, 2013. URL: <http://homotopytypetheory.org/book/>.
- [Wad89] Philip Wadler. Theorems for free! In *Proceedings of the Fourth International Conference on Functional Programming Languages and Computer Architecture*, FPCA ’89, pages 347–359, New York, NY, USA, 1989. ACM. [doi:10.1145/99370.99404](#).